SEMARAK ILMU PUBLISHING

Journal of Advanced Research in Applied Sciences and Engineering Technology

Journal homepage:
https://semarakilmu.com.my/journals/index.php/applied_sciences_eng_tech/index
ISSN: 2462-1943

Malware Detection using Deep Learning (DL)

Chowdhury Sajadul Islam[1], Madihah Mohd Saudi[2,*], Nur Hafiza Zakaria[2], Muji Setiyo[3]

[1] School of Science & Engineering (SSE), Department of Computer Science and Engineering (CSE), Uttara University, Uttara, Dhaka 1230, Bangladesh
[2] Cyber Security and Systems (CSS) Research Unit, Faculty of Science and Technology (FST), Universiti Sains Islam Malaysia (USIM), 71800 Nilai, Negeri Sembilan, Malaysia
[3] Center of Energy for Society and Industry (CESI), Universitas Muhammadiyah Magelang, Kota Magelang, Jawa Tengah 59214, Indonesia

ABSTRACT

The attack that occurred recently involved the utilization of malicious software, commonly referred to as malware, along with advanced techniques such as machine learning, specifically deep learning, code transformation, and polymorphism. This makes it harder for cyber experts to detect malware using traditional analysis methods. In view of the low accuracy and high false positive rate of traditional malware detection methods, this research proposes a fine-tuned deep learning model with a novel dataset that is compared with ANN, Support Vector Machine (SVM), random forest (RF), K-Nearest Neighbourhood (KNN) classifiers. Converting a malware code into an image could allow users to effectively identify the presence of malware, even if the original code is modified by the creator. This is due to the fact that the attributes of images remain unchanged, allowing for reliable identification. So, researchers used deep learning technology to detect malware, like detecting malaria from red blood cells. The deep learning model found that a more detailed analysis of malware data sets, focusing on RGB and greyscale images, is needed. These data sets currently rely on publicly available data, but the accuracy of the traditional model could be a lot higher. It also produces many false positive results. The main goal is to create a new data set and model using malware images to identify and categorize malware using deep learning without relying on existing image detection and transfer learning models. The researcher adjusted different hyper parameters, like the number of neurons, filters, stride, hidden units, layers, learning rate, batch size, activation function, optimizer, and epochs. Identifying and correcting issues in models, improving their clarity, stability, and fairness, as well as debugging and monitoring them, is a challenging task. To eliminate this obstacle, assess the model's behaviour and performance by employing various methods such as logging, profiling, testing, visualization, and model clarity. To overcome the challenges posed by the electricity shutdown, we utilized Google's cloud-based GPU and Python 3.7 language to conduct the experiment and train the model. Kali Linux is an operating system that can automatically encrypt file systems and has a lower chance of crashing the system due to malware in a virtual sandbox. The researcher used techniques like early stopping and cross-validation to prevent over fitting and assess generalization in addition to monitoring and evaluating the model's behaviour and performance. The researcher used different methods like L1 and L2 regularization, dropout, and batch normalization to improve the model's performance

* *Corresponding author.*
E-mail address: madihah@usim.edu.my

https://doi.org/10.37934/araset.57.1.253273

and avoid over fitting. The binary portable executable (PE) malware dataset is collected from Kaggle, Malimg, Virusshare, Malvis, MS Big2015, and VX-underground and finally converted to greyscale and RGB images to create a novel dataset to fill the lack of image dataset. The raw dataset was then rescaled to a 128x128 greyscale and RGB (red, green, blue) image and flattened to 1024-byte vector images input into convolution neural network (CNN) interpolation to extract features for malware detection. The newly acquired dataset is utilized as an input for the innovative DL algorithms in order to create a tailor-made model capable of accurately predicting malware. The findings in this customized CNN model by fine-tuning hyper-parameters with the three-channel RGB dataset outperformed a greyscale dataset with an accuracy of 98.7% and error rate of 1.3% in current malware compared with other artificial neural network (ANN), ML algorithms such as Support Vector Machine (SVM), K-Nearest Neighbour (KNN), and also Random Forest (RF) models. The proposed approach is compatible with all operating systems (Windows, Linux, and Mac) and can identify different types of malwares, such as packed, polymorphic, obfuscated, metamorphic, or variations of a malware family. There are some limitations to the shortage of malware image datasets and computational costs for fine-tuning hyper parameters in the whole model. Furthermore, the proposed approach detects malware and groups it into families without using common techniques like disassembly, decompilation, de-obfuscation, or running malicious code in a virtual environment.

Keywords:
Machine learning; Deep learning; Malware detection; Hybrid analysis; Malware applications (MA); Artificial neural network (ANN); CNN

1. Introduction

The first computer worm, known as Creeper, originally appeared on the DEC PDP-10's TENEX operating system in 1971. Rich Skrenta, a 15-year-old high school student, invented the first computer virus in history, known as ELK Cloner, which was spread through floppy disks in 1982. The Apple II OS is the primary target of ELK Cloner, which is regarded as the initial computer virus to increase in the wild. The Pakistani brothers Basit and Amjad Farooq Alvi designed The Brain, which is regarded as the initial PC virus, to target the IBM PC system. Malware means malicious software and is an umbrella representation that signifies a malicious program or code used for cyber-attacks to any extent since it is convenient to vulnerable networks and gadgets. Malware equips a wide attack vector for keyloggers, adware, spyware, worms, viruses, trojans, backdoors, cookies, and malvertising. It also appeals to irresponsible offenders who desire to assemble and disseminate malware in many directions with little disruption. According to SonicWall's 2023 Cyber Threat Report, described by Gulatas *et al.*, [1] in the first half of 2022, 57 million malware attacks were discovered, up to 77% from the same period in the previous year, as shown in Figure 1 below by Gulatas *et al.*, [1]. This six-month attack volume exceeds the annual attacks noted in 2018 through 2020. Despite several anti-malware tools and procedures, malware detection and analysis remain challenging because attackers constantly obfuscate the information used by encrypted polymorphic, obfuscation or modify their cyber-attack code to defeat more advanced security measures.

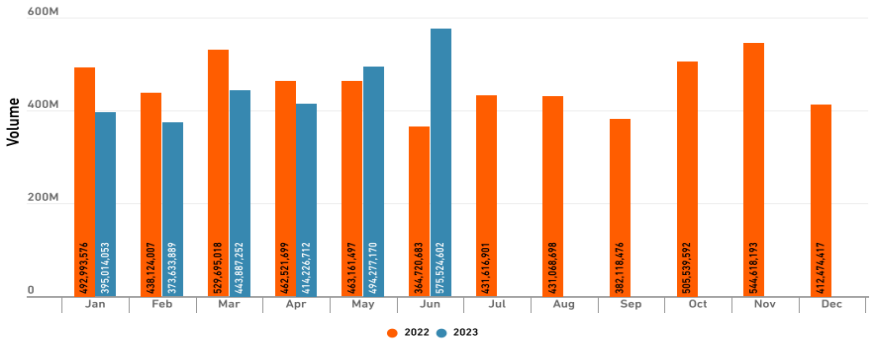


Fig. 1. Global malware volume report retrieved on 26 September 2023

In addition, several earlier techniques require better inadequate generalization to unidentified malware and scalability problems. Traditional malware identification technologies may be classified as either static or dynamic, depending on how the malware is currently being executed throughout the analysis process. The process of analysing the functionality and properties of malware using tools for analysis without running binary programs is known as static analysis-based identification. With a static analysis approach, malware feature sequences and feature code chunks may be located, and the working parts and flowcharts of each operational section can be acquired. Static identification has a relatively low cost, but it has a detection latency issue since it cannot identify malware that is unknown to it. The technique of using program debugging utilities to watch and follow malware while it runs is known as dynamic identification. Dynamic identification may identify malware quickly, but it also raises the cost of identification since it needs a virtual environment to function, and the malicious code has already infected any machines. Furthermore, most machine learning techniques for identifying malware have relied on characteristics retrieved from static or dynamic software analysis, necessitating debugging of code and implementation largely via offline processing; this makes the methodology's mostly because deep learning model performance in image classification has recently seen a boom in success, and image-based malware identification and classification are platforms agnostic. A deep learning-based method for identifying malware has been developed as a reaction to the shortcomings of conventional detection approaches.

New and undiscovered malware cannot be detected by conventional detection techniques since it relies on signature-based or dynamic analysis-based detection techniques. The convolutional neural network (CNN) is effectively used for image-based malware classification. CNN operates on raw data; this is one benefit over comparable image recognition algorithms such as face detection, object detection, etc. However, in malware detection, it is required to convert binary to image; a CNN functions as a feature extractor, which is highly useful as choosing features usually involves human specialists. Deep learning demands an enormous number of image datasets, which is a challenging task for collecting image-based malware data sets. The main motivation comes from the researcher's previous work, which was published in IEEE-EMBS, Malaysia IEEE Section. The paper titled “A Novel Idea of Malaria Identification using Convolutional Neural Networks (CNN)” is taken from previous research of CS Islam *et al.,* [2]. To determine parasitaemia, the goal is to identify and classify the unaffected and damaged blood cells in an impure blood sample. Since traditional high-end light microscopes have lenses that are of poorer quality, traditional algorithms are unable to analyse these photographs; instead, the author utilizes a pixel classifier framework in the Convolutional Neural Networks (CNN) to prioritize the white blood cells. This paper also researches on malware classification, where the malware image pixels are used for feature extraction. The X-Layer Optimization in CRN Using Deep Q-Network for Secure High-Speed Communication is taken from the previous studies CS Islam *et al.,* [3] of this paper writer researcher proposed end-to-end safe communication for 5G and IoT traffic, the CRNs use cross-layer (X-layer) enhancement of the OSI model by determining the parameters from the physical layer (Layer-1) and the network layer (Layer-3). The proposed approach uses a Deep Q-network (DQN) to pick the next hop for transmission depending on the duration of the wait, in accordance with the innovation, which is dependent on the bit error rate used for transmission and the packet loss rate for secure encrypted high-resolution video traffic over 5G networks, is a benefit of neural networks (NNs). The above research influenced the researchers to use CNN and DL to detect malware from image-based datasets because if blood cells and frequency are possible to detect using DL, why not malware detection. The above articles motivated me to do the following:

i. Comparing three-dimensional and one-dimensional CNNs to address the mislabelling issue of multiple image-based malware data sets on the proposed novel dataset.
ii. The viability of utilizing CNN parameters fine-tuning techniques to evaluate the performance of malware detection.

It is observed by researchers that lots of research work is done on DL by attaching other techniques or only DNN, but this research shows that only using DNN by proper tuning of parameters with newly built real datasets can achieve good results compared with existing work. There needs to be a more comprehensive analysis of RGB and grayscale image-based malware datasets, and the accuracy is not so high and generates lots of false positive rates in traditional CNN. The gap of the current malware image-based (RGB and grayscale) dataset is out of current malware images when training the deep learning (DL) model and it has failed to detect a few malwares. All is known about the value of the convolution and polling layers in DL; otherwise, the user can use any dataset but will not detect accurately, as a human eye can identify an ant in their skin. There are blended malware data sets in Kaggle, malevis, and mailing, with all being either grayscale, RGB or only binary files. So, researchers are trying to find a way to solve this problem by developing a novel dataset and compared with different models to ensure the ethical accuracy of the dataset. Model fitting with grayscale and RGB is a challenging task because there are multiple families of malware for four channels (1 greyscale, 3 RGB). The same configuration of parameters does not work for both datasets in the same traditional model. The parameters, hyperparameters, and adding new layers, such as attention to fit a new model, require more time to adjust than a traditional model. To solve the issues for the requirement of large memory to tune all parameters in many hidden layers with a large dataset, firstly, the researcher builds a small model with a small balanced dataset and then increases the number of families of datasets and tunes all parameters gradually with adding hidden layers in DL to build this model, though it requires more time. This research included thorough experiments and analysis of the proposed approach using a collection of malware-augmented images in both grayscale and RGB. Using training data accessible for every procedure, first, use a basic 2D CNN for grayscale images. Employing a 3D CNN (in which an input is an accumulation of instances over a time interval) for RGB images improves the performance of the CNN classifier. This process significantly reduces the number of mislabelled data samples during training and data collection. The authors conduct tests using data gathered from infected individuals with different malware, like rootkits, trojans, ransomware, spyware, etc., which are used in tests chosen at random for ethical reasons. Then, the model extracts the suitable features for the model from each malware image file. These features are demonstrated as inputs to a deep-learning classifier. Here, the authors utilized two types of classifiers: one is a multi-class classifier for malware family classification, and the other is a binary classifier to classify between benign and malicious files.

The challenge with the existing image-based dataset is that it is not updated with current malware and is unbalanced, so it is crucial to extract novel features from diverse sites to generate a novel image dataset. The balanced dataset is obtained by augmentation and setting the parameters of algorithms matching the model. The foremost objective of this paper is to detect malware with heightened accuracy and low loss of novel image-based RGB and grayscale datasets using deep-learning models by tuning parameters automatically and manually, which requires a balanced dataset to avoid overfitting problems. To balance the dataset, the researcher applies horizontal flip augmentation and tests the quality by using the model to ensure no biased data. It happens when a model grows overly complicated and picks up on the quirks and noise of the training set, which results in subpar performance on untrained data. Regardless, the authors used an automatic fine-tuned function from tensor flow and finally utilized manual tuning to adapt the layers and hyperparameters

with the augmented dataset. The outcome of the proposed model is more satisfactory than the previous model, as seen below in Table 1. The methods applied for PE to image conversion here are binary-to-image (B2I) transformation algorithms for malware PE files (.apk,.acm,.mui,,.dll,.efi,.exe,.tsp,.sys) to transform the image to create a new dataset. The authors used horizontal flip augmentation to balance the dataset and tensor flow function for auto-tuning parameters used for fitting the multilayer model, eventually testing the binary classifier later. Analogizing the results with other datasets and models to validate the suggested model, a novel set of data presented DL algorithms got descending loss with high accuracy.

The efficiency of the proposed fine-tuned CNN model was compared with many non-fine-tuned traditional CNN-based models and ML models using different ratios of data set splitting for training and testing with the proposed image-based datasets. When it comes to evaluating the model with different statistical values, which are accuracy, precision, recall, and the F1-score, the tuned CNN approach outperformed the traditional non-tuned CNN-based models in both data sets while comparing with the experimental result. The significant thing is that most of the methods show constant performance on all training and testing data, which makes the proposed novel dataset and the novel optimized CNN model's work better than other models. The efficacy of the multiclass classifier with the recent raw malware dataset (36551 samples) created by the author and also evaluated using several classification metrics involving the train-test split approach (80:20 ratio) and 10-fold cross-validation was then applied from the dedicated multicast model to the binary model for benign and malware prediction. The binary classifier test dataset contains 5813 benign images and 16550 malware images from the novel dataset, which are not used to train the model. The 10-fold cross-validation method involved in the presented framework in RGB features delivers a better multiclass classification rate (98.7%); in contrast, the grayscale offers 95% accuracy in the CNN but has poorer accuracy achieved in other ML models. The proposed technique minimizes the gap in malware detection using deep learning with a novel dataset. The following are this paper's primary contributions:

i. Build RGB and grayscale image-based data sets to train a malware classification DL model.
ii. To propose a customized deep learning model for malware detection based on image-based malware classification with high accuracy compared with others by fine-tuning parameters and hyper-parameters.
iii. A comprehensive evaluation of the proposed model’s capacities and constraints based on the Confusion Matrix (CM) with the prior studies delivers practical insights into its significance for real-world performance.

The remainder of the paper is organized into different sections: the first is an introduction that describes the problem statement, the objective of the paper, the method used and the findings; section two illustrates the background, which explains the latest literature on the topic; section three expresses methodology, which illustrates the overall tools and techniques used in the experiment; section four presents results and discussion comparison with others work elaborated with details explanation of fine-tuning, and section five organizes the conclusion which concluded with limitations and further works.

2. Background

The capacity of evasion tactics for threat analyses and malware discovery illustrates in this research. In static and dynamic malware analysis, extracted features will be further used for malware

classification model by machine learning algorithms and neural networks. CNN was laboriously utilized in image analysis by Smmarwar *et al.,* and taken from the previous studies [4] to construct the model with beneficial results. To overwhelm deficiencies, artificial neural network (ANN) approaches, CNN involved the proportion of weight, convolution layer, pooling, and size of kernel techniques mentioned by Asam *et al.,* are taken from the previous studies [5]. The efficiency and cost of the malware visualization method have recently been investigated. Analysis of harmful code effects, which are taken from past research, is the main focus of conventional MD techniques introduced by Conti *et al.,* [6]. To detect unknown types of damaging malicious code, these capabilities further benefit cutting-edge ML-based MD approaches. These systems, however, were unable to identify fresh virus variants. A model based on images was proposed and taken from the previous research of Venkatraman *et al.,* [7] to identify malware. The paper is taken from the previous studies of K. Shaukat *et al.,* [8] which offer a novel DL technique for malware identification. It combines static and dynamic analysis to produce better results than traditional methods. The research is taken from the previous studies of Akhtar M.S *et al.,* [9], which demonstrated the detection of harmful traffic on computer systems; the effects of using machine learning (ML) methods for malware analysis and identification to calculate the integrals are of the variations in correlations symmetric (SVM, Naive Byes, RF J48, and methodology). A novel malware classification approach employs double attention-based CNNs in images taken from the previous research of Alnajim *et al.,* [10]. The research taken from the previous studies of Djenna *et al*., [11] suggests classifying and detecting five categories of contemporary malware by combining dynamic DL techniques with heuristics techniques. Through an accuracy rate of 98.17%, taken from the previous research of Wolsey *et al.,* [12] an overview of cutting-edge AI methods for malware identification and avoidance offers a thorough examination of the most current research in the area. A reinforcement strategy that uses ML to recognize different kinds of malware and their different forms was first released and taken from the previous studies of Liu *et al.,* [13]. The research is taken from the previous studies [14] of Naeem *et al.,* who developed a commercial Internet of Things (IoT) malware identification method. The writers created a sniffer gateway to keep track of traffic in and out of log files. The previous research is taken from Awan *et al.,* [15], who introduced malware categorization founded on images. The VGG-19 network used by Sharma et al., is taken from the previous studies [16] to categorize 25 well-known malware images. An MD framework based on Xception CNN for malware picture classification was proposed in a related paper. The authors claim the models outperform the current frameworks in terms of output. An MD framework was created by Yadav *et al.,* [17] utilizing Android malware images. Using Java byte code, suggested by Obaidat *et al.,* is taken from the previous studies [18], a CNN model for malware detection. Bi-directional DL was created by Chaganti *et al.,* and is taken from the previous studies [19] to categorize IoT malware photos. A different study conducted by Falana *et al.,* [20] produced a method to turn binaries of PE malware into a picture to aid in classifying malware. CNN models have a typical accuracy of 96.77% in identifying major viruses by identifying even minute alterations in an image. Nevertheless, it has several drawbacks, such as an elevated rate of false positives that incorrectly label many lawful actions as pestering and necessitate further data. Besides, the current techniques require advanced computational resources to produce a successful result. With an astonishing accuracy of 98.70% in malware identification, the suggested approach has proven to be incredibly effective of 98.70% on the Malvis, Malimg, and MS Big2015 benchmark datasets taken from the previous studies [6,13,17,20] and research work of Zhu *et al.,* are taken from the previous studies [21]. The comparisons of the current malware detection (MD) frameworks with the proposed model are listed in Table 1 below.

Table 1
Comparing with related work with proposed work regarding dataset, methods, and accuracy

References No.	Dataset	Method	Accuracy (%)
[3]	Malimg	DB feature classifier	Average: 98.6
[4]	Windows mal bin	Conv. Siamese neural network	Average: 98.5
[6]	Malimg	ML, SVM + CNN	99.06
[7]	SMOTE, IoT	Naive Bayes, SVM, J48,RF, CNN	Average:81.54
[8]	Malimg	Deep Learning (CNN)	Average:98.95
[9]	CICAndMal2017	Behaviour & Heuristic based DL	Average:94
[11]	Microsoft BIG	CNN and GAN	Average: 6.25
[12]	Leopard dataset	Deep CNN model	Average: 97.5
[13]	Malimg	Spatial attention CNN	Average:97.62
[15]	Android malimage	SVM and Random Forest	Multi:92.9, bin:100
[16]	Java byte code	CNN	Average: 98.4
[17]	IoT malware	Bi-directional DL model	Average: 98
[18]	Malevis, Malimg,	Deep GAN and CNN	Average:96.77
[19]	G. Play Virus share	Deep Learning (CNN)	Average:96.9
[20]	Malimg	CNN	Average 98.4
[21]	IoT bot net Attack	ML Algorithm including RF	AdaBoost98.87
Proposed DL Model	Novel dataset	DL with 3-channel RGB & greyscale data	Average:98.70

Yerima *et al.,* are taken from the previous studies [24] presented a semi-supervised SMS spam detection procedure by devoting a chi-squared selecting strategy with a 98% accuracy rate. The OCSVM and SIMCA single-class classification prototypes were presented by Kelis *et al.,* are taken from the previous studies [25] using the OCSVM classifiers. The inspection outcome demonstrated the two models' comparative forecasting success percentages for benign data, which were 78.8% and 86.9%. An anomaly-based and signature-based network intrusion detection system was created by Alazzam *et al.,* are taken from the previous studies [26], and it outperformed other strategies in identifying zero-day assaults and reducing the number of missed alerts. The "KDDCUP-99" dataset was utilized in this investigation's training stage. The Pigeon Inspiration Optimization (PIO) was operated to determine the most reasonable features of training sets. In the presented approach minimized false alarms and enriched detection rates by combining the two subsystems simultaneously to assess every packet. A novel procedure for using data from Phasor measurement units (PMUs) to determine clever grid attacks was presented by Nguyen *et al.,* [27]. It is predicated on semi-supervised anomaly recognition and influences publicly open datasets on electric power hacking. Four supervised and four semi-supervised algorithms corresponded in terms of efficiency. The combo classifier WOC-SVM-DD enriched weight computation technique and also attacked determined samples with large-scale categorization was achievable and successful Zhao *et al.,* are taken from the previous studies [28] reached test performance with 99.3% accuracy. The HIDS model was assembled and assessed by Khraisat *et al.,* are taken from the previous studies [29], which were considered by operating the ADFA and NSL-KDD datasets. To gain high-accuracy results, the researcher carried out three evaluation steps. The RBF kernel was utilized in the final step, resulting in detection for the NSL-KDD dataset of 72.17% and the dataset ADFA of 76.4%. After creating an anomaly-based NIDS, the authors Nguyen *et al.,* are taken from the previous studies [30] presented a transformation to raise the detection rate of intrusions by utilizing only the KDD99 dataset's typical class. They employed a nested procedure to determine the ideal hyperparameter to gain accuracy rates of 98.25% and the most periodic false rates of 12.5%. The researchers Min *et al.,* are taken from the previous studies [31] offered the memory-augmented auto-encoder (MemAE), an unsupervised learning model. All three models were trained on benign data using CICIDS 2017, NSL-KDD, and UNSW-NB15 datasets.

The OCSVM model accuracy rates were 81% on UNSW-NB15, 76% on CICIDS 2017, and 94% on NSL-KDD questionnaires. Accordingly, MemAE was implemented to fix the auto-encoder extreme generalization issue. A zero-day attack detection system based on anomaly-based NIDS that employs unsupervised approaches was also instructed by Verkerken *et al.,* are taken from the previous studies [32].

Further, PCA simultaneously functioned well in variety, with a multiple observed AUROC of 84%. Furthermore, a cutting-edge NIDS model engaging in machine learning algorithm procedure was constructed and evaluated by Mahfouz *et al.,* are taken from the previous studies [33]. The suggested method relies on deciding consistent traffic. A contemporary analysis with a honey network fetched a 97.61% accuracy rate. For IDS, the authors of Binbusayyis *et al.,* are taken from the previous studies [34] proposed an unsupervised, more profound learning technique. The recommended classifiers were merged into the 1D-CAE procedure into the collaborative enhancement structure, and the dataset UNSW-NB15 and NSL-KDD have been placed into training. CNN with auto-encoder approaches was impacted for both datasets to create influential feature visualizations. Operating the NSL-KDD dataset enriches OCSVM's capability to indicate 91.58%, while with the UNSW-NB15 dataset; it reaches 94.28%, as demonstrated in Table 2 below. But in this research achieve accuracy 98.70%.

Table 2
Comparing of proposed techniques with others malware detection techniques regarding dataset, methods, and accuracy

References No.	Dataset	Method	Accuracy (%)
[22]	SMS Spam collection	chi-squared	98.00%
[23]	Cassava starch samples	PCA	86.90%
[24]	KDD CUP-99	Cosine PIO	99.70%
[25]	power system samples	DAE, PCA	86.00%
[26]	banknote authentication samples	Manual multi techniques	99.30%
[27]	ADFA	Manual with parallel	76.40%
[28]	KDD CUP-99	OCSVM nested	98.25%
[29]	NSL-KDD	Manual	94.00%
[30]	CIC-IDS-2018	Manual	97.00%
[31]	honey network	Manual	97.61%
[32]	UNSW-NB15	AE, CNN	94.28%
Proposed	Novel dataset	DL with 3-channel RGB & greyscale data	98.70%

3. Methodology

This methodology section illustrates the methodology that will be used in this research work. The usability of model work flow is given in Figure 2.

The research methodology includes the collection of data, how the data was analysed, and the development of models to detect malware accurately. The results are then examined in the context of recent research in the area of malware identification. This section describes the classification of malware that makes use of CNN compared with ML and ANN. To categorize and detect malware-based cyberattacks, researchers propose the deep learning architecture for binary classification and multiclass classification.

Before using the proposed technique, the malware executable files are first converted into images. Then, the proposed fine-tuned Convolutional Neural Network (CNN) model is used to identify and classify malware families from the converted image data. By doing this, researchers learn about the many malware variants and their behaviours. After finishing data processing, section 4.1

illustrates the parameters and parameters tuning, which required additional computational capability during the training and testing period. So, a high-configuration computer with high-speed internet takes less computational time, which is related to accuracy. The experiment was conducted by the following configuration of hardware and software, which are shown in Table 3.

Table 3
The Hardware and software's requirements for training and testing

Hardware	Lenovo Laptop
	Processor: Intel Core i5 13 Gen, GPU: Get Force RTX-3060 GPU, Nvidia's GA106 silicon, 3840 CUDA cores. Memory: RAM 32GB Storage: NVMe SSD
Software and platform	Google's cloud-based GPU, Kali Linux, Python 3.7 programming with libraries including Tensor Flow, Keras, and Sykit Learn
Dataset	Kaggle, Malimg, Virusshare, Malvis, MS Big2015, VX-underground

The proposed deep neural network (DNN) is an artificial neural network (ANN) that has many hidden layers which can automatically learn features, extract features, and train convolution neural networks (CNN) to detect malware. The researcher's proposed model will analyse and detect malware PE file codes and enable the automatic detection of unknown real-time attacks before exploiting the malicious codes on the target devices. Training deep learning models is a challenging and expensive task; it requires efficient hardware with suitable software for coding with an expert programmer. Adequate equipment such as CPU, GPUs, TPUs, RAM, NVMe SSD, and distributed computing facilities can significantly improve training, testing speed, and efficacy of the model. As technology evolves, envision seeing ever-more cutting-edge hardware and software tools, enabling the training of ever-more-powerful artificial intelligence, or DL, models in the future. NVIDIA's CUDA platform and the cuDNN package allow GPU-accelerated deep learning image-based training and testing till now in local devices.

It is mentioned that the researcher made a novel dataset that required the collection of PE malware files from six sites, a few of which are already converted to two types of images (RGB, Grayscale) datasets to train the DL model. Most of the binaries of PE malware were collected from Kaggle, Malimg, Virusshare, Malvis, MS Big2015, and VX-underground, which require conversion in an image file, and the author used a laptop GPU for faster conversion. Google Co-lab was used due to the power supply problem in the researcher's country. It can be seen that it finished training (90-99%), but the power is shut down, and UPS cannot take too much GPU load. Then, it would require restarting the training process from the beginning or from a certain point where the GPU finished a few works. This is a specific reason behind choosing Google's cloud-based GPU for its uninterrupted power supply and high speed, but there are security and privacy issues in the cloud. All know that though the PC power is shut down but co-lab continues the training process, it will not stop; it will automatically continue on the server even though the client is shut down. The researcher's local GPU laptop used Python 3.10 for converting PE to image, but Colab supported Python 3.7 with TensorFlow and Keras libraries at the time of training, and some libraries were also supported in 3.7. It is possible to create virtual environments for current Python 3.12.1 on a local laptop, but it is not possible in the Google Cloud environment. The author chose Kali Linux because the researcher normally uses it for penetration testing, the file systems are also encrypted by default, and there is less chance of malware infection from the sandbox to Linux due to the encrypted file system.

3.1 Dataset Collection

To detect malware, researchers feed the image data in CNN by neurons (pixels are also used as tensor TF algorithm) exchange and fine-tuning parameters (bias, weights), specific hyper-parameters (number of neurons, kernels/filters, stride), learning rate, minimum batch size, and number of epochs. The model-specific hyper-parameters (number of hidden units, first layer, number of layers), activation function, optimizer, and clusters in K-means). Deep learning requires an enormous number of datasets for training, but there are a small number of recent image-based malware datasets available, and there are also difficulties that datasets need to be balanced for training and testing. Therefore, the proposed model operated a newly assembled binary-to-image dataset with 36,551 samples of 34 malware families, such as adware, botnets, spyware, malspam, exploits, etc., described in [4,15,18,19]. It is operated for investigation of the raw PE byte stream to the image that has been re-scaled from 32x32 to 128x128 grayscale and RGB image using the Nearest Neighbor Interpolation (NNI) algorithm for image processing and flattened to a 1024-byte vector. The purposes of the new dataset are

i. PE byte plot image classification
ii. managing RGB and grayscale byte plot images in the two different datasets.

The researcher's proposed approach is split into six primary parts, as shown in Figure 2, where the output of every step performs as the input to the next step in the process.

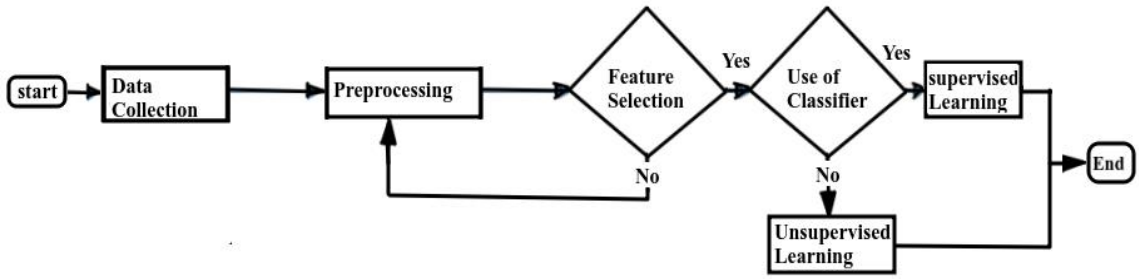


Fig. 2. Process flow diagram of deep learning-based malware identification

3.2 Data Conversion Techniques

Using the RegEx pattern finding in Python 3.7, convert raw PE to RGB, and grey-scale image files, saved in .png format, are shown in Figure 3 below. Here, many simulations were done to determine the best percentages for training and testing sets for effective malware detection with high accuracy. The findings revealed the ratio (80:20) of the image dataset achieved the suggested and superior efficiency compared with the other ratio for fine-tuned CNN algorithms.

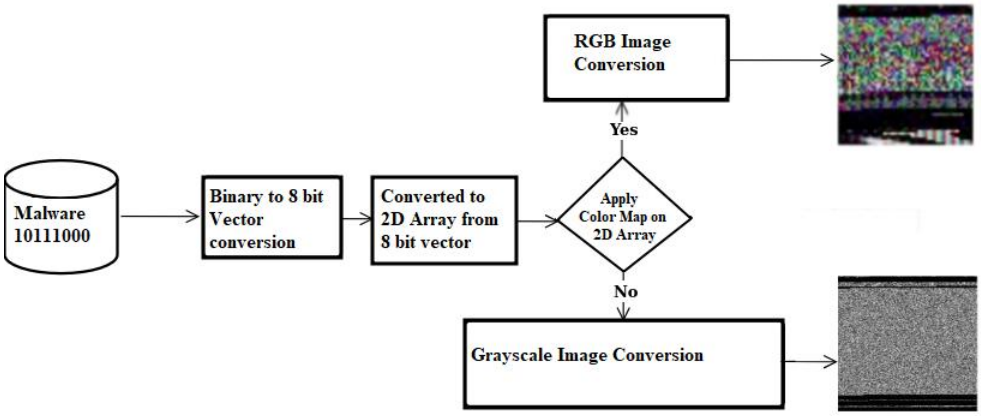


Fig. 3. PE binary file converted to a grayscale and RGB image data set

3.2.1 Data pre-processing

The primary goal of transforming PE malware into grayscale and RGB images using a B2I conversion algorithm to assemble two-dimensional images is to extract additional characteristics and information that cannot be acquired or gathered from the original malicious PEs in binary representations. As an outcome, transforming the feature set to grayscale and RGB for visual representations eliminates the requirement of reverse engineering PE file methods and also does not require any specialization knowledge for analysing data. The proposed model's classifier identification capability performs effectively on diverse image optics, augmentation, and representations. Because the RGB picture was $n \times n$ size, the byte code image likewise required sampling to obtain a grayscale image of $n \times n$ size. Figure 4 shows a sample of a 512×248-byte grayscale image that transformed into 64×64 and 128×128 input images after piece and fixing 128×128 size images for input.

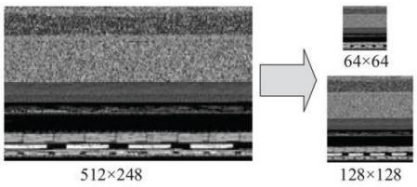
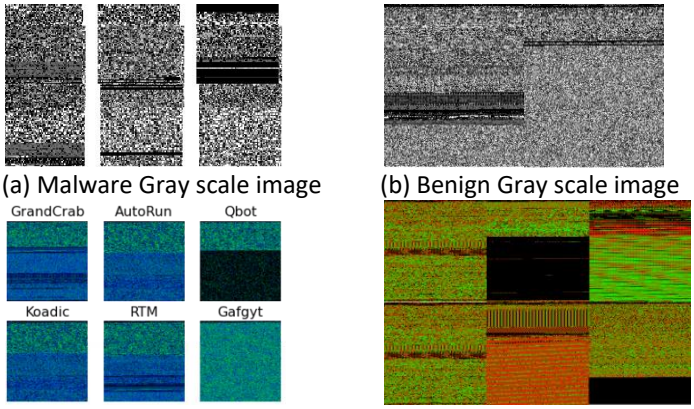


Fig. 4. Reshape the byte code image into a predetermined size

As shown in Figure 5, benign and malicious sample image datasets have various structures, types, and shapes. The relationship between the sizes and the widths of the produced image representations is shown in Table 4. Because of the striking contrasts in the graphical properties of the benign and malware, authors were motivated to modify and utilize standard DL-based CNN algorithms for malicious applications' safety. The colour and grayscale images are scaled before being sent to a fine-tuned CNN model to automatically extract features, training, testing, and classifications.



(a) Malware Gray scale image (b) Benign Gray scale image

(c) Malware RGB image (d) Benign RGB image

Fig. 5. Malware and benign PE files 2D graphic representation

3.2.2 Influences of image size categorization on model performance

The proposed technique forms *n* key byte codes; the value of *n* is essential to implementing the presented model. When *n* is equal to 16, 32, 64, or 128, the image size is 16 × 16, 32 × 32, 64 × 64, and 128 × 128 through testing acquired multiple sample classification results in the combination of *n*-sizes in Table 4. When *n* was more than 128, performance did not increase in association with *n*, and the time overhead overgrew. As a result, the *n* must be 128 for a lower computation time.

Table 4
Influences the efficacy of classification of various sizes of images in the model

Image size	Accuracy		Precision		Recall		F1-measure		Time (ms)	
	Gray	RGB	Gray	RGB	Gray	RGB	Gray	RGB	Gray	RGB
16 × 16	0.8954	0.9154	0.8941	0.9037	0.8825	0.8934	0.8865	0.9001	14	12
32 × 32	0.9432	0.9641	0.9421	0.9630	0.9524	0.9672	0.9501	0.9661	28	30
64 × 64	0.9703	0.9805	0.9532	0.9871	0.9600	0.9833	0.9600	0.9862	42	42
128×1288	0.9708	0.9887	0.9531	0.9846	0.9621	0.9812	0.9623	0.9829	177	180

3.3 Normalization

Feeding the dataset into a DL model requires normalization because various dimensions are produced from various PE files, which are challenging to categorize. The images were re-sized to 128×128 pixels to solve this problem.

3.4 Identification of Attacks using CNN

CNN can identify malware using features like human eyes to find spotlights to separate objects. To construct a customized multilayer and binary CNN model with fine-tuning from the traditional convolution layer to the fully connected layer, weight, pooling, dropout, and kernel are required to adjust properly. The proposed framework of the whole process is displayed below in Figure 6. The most popular model form in Python programming is the sequential type, and Keras is the simplest method for creating a layer-by-layer CNN model. To build a multiple-layer CNN, with an input layer size of 512 and a kernel capacity of 34×73024, CNN started modelling. An initial convolution layer is present, with a batch normalization of 128 and a data input number of 73024.

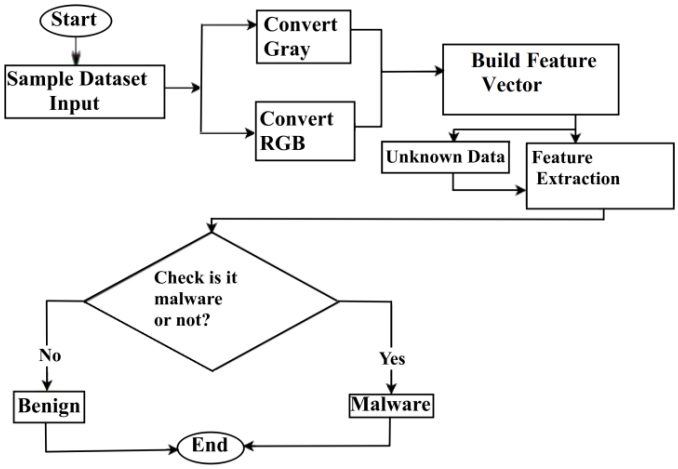


Fig. 6. Flow diagram of malware detection using CNN

The max-pooling layer takes an input of 128, and a batch normalization size is 32. The additional conv-layers are constructed with the filter (kernel) size of 34 × 3843 and a maximum 64 number of pooling layers. The ReLU activation function is carried out sincerely along with dropouts to obtain the 34 × 1 output. Applying the optimization algorithm Adam and the loss function "binary_crossentropy ()" within the uniform kernel output layer can optimize the classification label to produce a predicted class label. The activation employed is called SoftMax, which provides probabilities for every category that add up to 1. The model will adjust its forecast based on the class with the highest likelihood. The model's parameter overview is shown in Table 5 below.

Table 5
CNN model parameters configuration

The parameters of CNN	Shape	Size
Conv2d	34×73024	512
Batch Size	–	256
Max Pooling 2D	–	128
Conv2D	34×3843	256
Batch Size	–	128
2D Max-Pool	–	68
Flattens	–	256
Flattens	–	128
Dense layer (ReLU)	–	68
Dropout amount	–	0.31
Dense layer(ReLU)	–	16
Rate of Learning	–	0.00001
Decay rate	–	0.001
Optimizer used	–	Adam
Batch size	–	13
Epoch rate	–	10
Loss factor	–	Cross-entropy (Binary)
Activation Fun.		SoftMax
Metrics	–	Accuracy
Output Layer	34×1	2

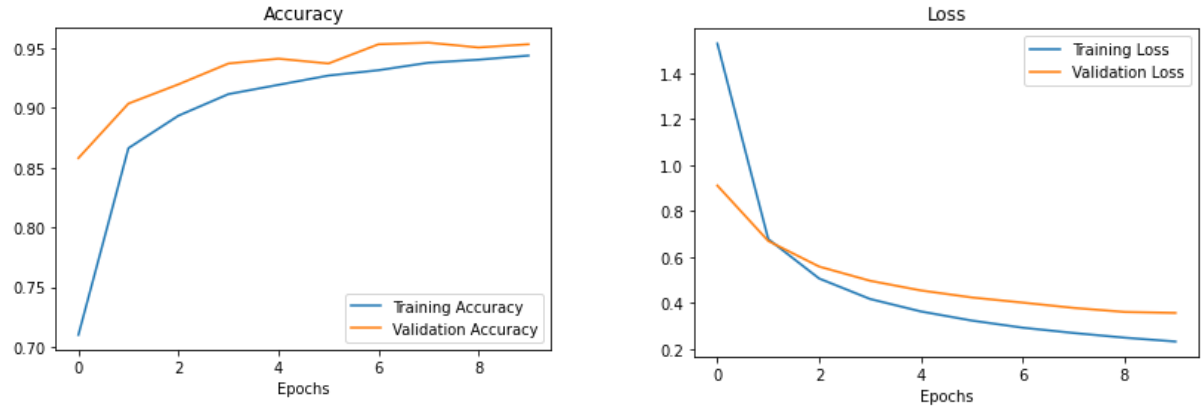
3.5 Implementation

To implement the model, upload a customized dataset to Google Drive and connect with Google Colab notebooks to access the dataset. The Python 3.7 language is used for coding to visualize and

normalize data, and the DL model applies for training, testing, and validation over days. The confusion matrices are used to evaluate the accuracy of the models. The epochs for the dataset were set to ten for testing purposes after fine-tuning the hyperparameters. Initially, the proposed model dataset had an unbalanced distribution of malware and benign PE files despite the author's use of horizontal flip augmentation to balance the data set.

4. Results and Discussion

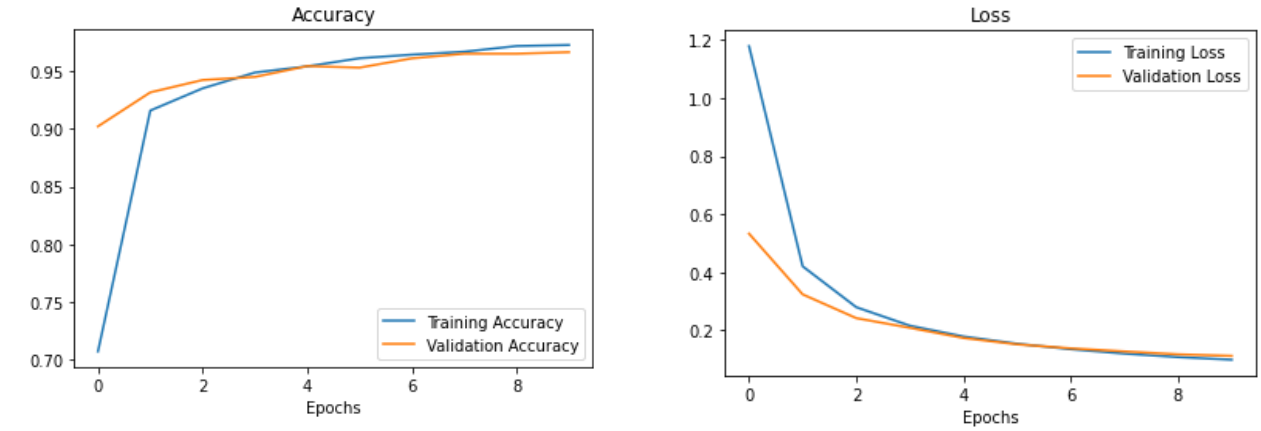
There are two different types of datasets: one is multiclass, and the other one is binary used in this study. With the experiment's grey-scale data set, it is displayed in Figure 7 that the recommended model's validation accuracy is low, though its loss is considerable but not acceptable. Still, RGB got higher accuracy with low loss, as shown in Figure 8. It is also observed in Figure 7(b) that the loss curve is raised for grey-scale images, whereas the loss for RGB images is near zero, as shown in Figure 8(b).



(a) The model's training accuracy and validation accuracy (b) The model's training Loss and validation Loss

Fig. 7. The model's training accuracy, validation accuracy, and loss for grey-scale image

Using a fully balanced dataset and minimizing the kernel and stride sizes makes it possible to decrease the loss. However, it will require additional computing time; accuracy will become higher. When using balanced colour images with ten epochs, the accuracy of the training dataset and testing dataset's accuracy is better in the proposed tuned model shown in Figure 8.



(a) The model's training accuracy and validation accuracy (b) The model's training Loss and validation Loss

Fig. 8. The model's training accuracy, validation accuracy, and loss for multiclass RGB image

It observes that the accuracy of the testing process and the generated loss for balanced the images of RGB and grayscale stood steady behind a trimmer than seven or nine epochs rate. Confusion matrix (CM) is depicted in Figure 9, where statistical evaluations visualize and summarize the performance of 34 classes of malware families.

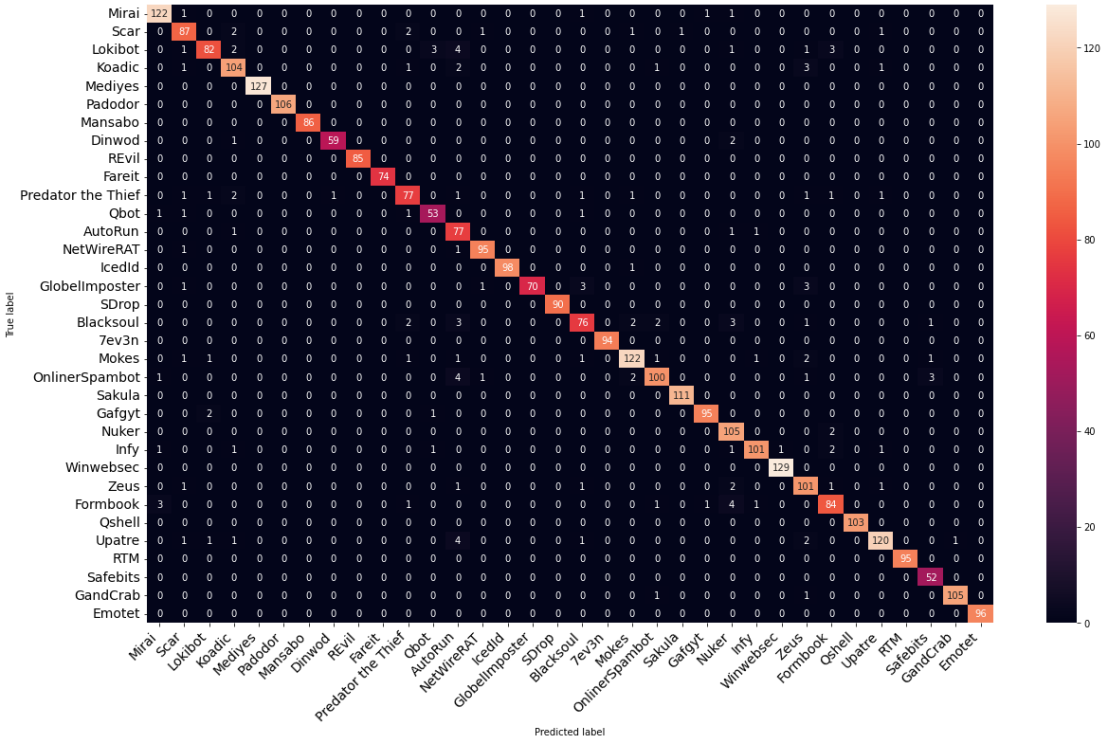


Fig. 9. The confusion matrix of the model which proposed for the classification

As a result, it can be seen that for RGB images, the CNN algorithm produced good detection effectiveness at a lower number of iterations (epochs). Therefore, malware attacks are strongly encouraged to be recognized precisely and quickly in cyber security applications using a three (3)-channel RGB image dataset.

4.1 Tuning the Model

The sampled dataset contains many imbalanced benign and malware images, converting the balanced image size using a horizontal flip augmentation technique. Therefore, over-fitting was avoided as much as feasible throughout the training and testing procedures. Compared with other ANN or CNN algorithms, the proposed DL delivers the most remarkable and best detection for benign and malware images. As a result, this research investigates the architecture, conducting of training, fine-tuning the parameters, hyper-parameters, and layers, properly adjusting optimization function, and achievement of identification outputs. All Convolution and Max Pooling layers utilize a stride value of 2×2. The ReLU activation function speeds up the training procedure. The most sumptuous benefit of the suggested fine-tuned CNN model for grey-scale and RGB datasets over default CNN algorithms is that it may be enhanced, as its layered sections feature intrinsic repeating different kinds of layers that are simple to modify and update. Furthermore, the presented fine-tuned algorithm enriches the performance of detection without requiring more deep training, where the consisting convolution or separable convolution layers utilize separate kernels that can identify and obtain distinct texture-like characteristics in benign and malware images with just a few training

repetitions. As a result, it is highly effective in computing and attractive for use in detecting PE malware attacks.

Aside from utilizing the benefits of the presented DL model, hyperparameters used in the deep learning model were tuned to adjust appropriately. This fine-tuning outperforms methods based on accurate detection versus superficial tuning with inferior computing difficulty versus fine-tuning in the DL model. As a result, in the DL approach, all the hyperparameters applied in DL algorithms are refined and fine-tuned as long as the effectiveness with a heightened identification is reached. The conclusive fine-tuning and optimizing variables involved in suggested image-based DL models are as follows: speed of learning 0.0001, optimizer used as Adam, the regression used for linear regularization (L2 - Reg.) in weights decline of 0.0011, the highest possible epochs numbers corresponding to ten, the smallest size of batch quantity is 13, the dropout rate is 0.51, learn rate schedule variables set to "piecewise" learning rate drop priority such fine-tuned the hyperparameters have been carefully selected to keep away from over-fitting and maximize the training and validation process' efficiency. In addition, the designed DL model also employs the back-propagation approach to tune for enhancing hyperparameters as neuron weight associated with the proposed DL model, which was developed at the beginning to accomplish higher identification efficacy for recognizing malware attacks. The model suggested outperforms the Microsoft Big 2015 and the Malimg datasets and transfer learning models used, as shown by the results in Table 6.

Table 6
Comparison of different dataset accuracy of various models with proposed model and data

Models	MS Big 2015	Malimg	Novel RGB Dataset	Novel Gray Dataset
Proposed DL Model	96.65%	97.65%	98.70%	97.08%
ANN Model	94.45%	95.34%	95.13%	94.46%
Support Vector Machine	94.99%	94.53%	95.13%	95.76%
K-Nearest Neighbor	94.31%	95.68%	95.12%	93.67%
Random Forest	94.01%	95.31%	95.92%	93.54%

In the binary classification process, both malicious and benign files are used in this detection to observe how it turned out. The new data set accuracy and loss performance compared with the various models are shown in Figure 10 below. Additionally, there is a comparison between the new grayscale and RGB datasets, as shown in Table 7, where the accuracy of the RGB dataset is higher than that of the grayscale dataset.

(a) The model's training accuracy and validation accuracy (b) The model's training loss and validation loss

Fig. 10. Accuracy curve of benign and malware classification

Table 7
Comparison of RGB and grayscale dataset in binary classification on various models

Models	Accuracy in RGB	Accuracy in Gray
Proposed DL Model	99.87%	96.78%
ANN Algorithm	97.13%	96.12%
SVM	96.93%	95.36%
K-Nearest Neighbor (KNN)	97.76%	95.87%
Random Forest(RF)	97.96%	96.39%

The experiments conducted verify the feasibility of the suggested image-based technique by analysing two cases of classification of binary classifiers. To demonstrate the effectiveness of the newly developed balanced dataset using data augmentation techniques applied on fine-tuned DL models such as ANN and CNN, these two classification scenarios, where RGB performs better, are examined in Table 8.

Table 8
Comparison Matrices between CNN and ANN with grayscale and RGB dataset

Model	Accuracy (%)		Error Rate (%)		Precision (%)		Recall (%)		F-1 Score (%)	
Name	RGB	Gray	RGB	Gray	RGB	Gray	RGB	Gray	RGB	Gray
ANN	85	83	15	17	87	82	99	97	93	89
CNN	98.7	96	1.3	4	80.7	98	98	96	89	97

These extensive tests have been conducted on the proposed DL validation, its heightened identification rate, and excellent classifier effectiveness in various RGB malware images. The accuracy of the proposed model can increase if a balanced data set is used. It looked at the model's accuracy using the maximum n features from five to twenty-five. It found that utilizing just the total fifteen times could achieve the model's accuracy on the RGB (98.7%) and greyscale (95%) characteristics of novel malware.

5. Conclusion

The researcher developed and validated a novel strategy to automatically detect and segment binary PE to an image-based malware dataset using a fine-tuned deep-learning model. To achieve this goal, the researcher proposed a novel malware dataset that converted to grayscale and RGB image datasets. Then, it applies customization techniques by fine-tuned DL model where all hyper-parameters are not only tuned using the default tuning function of tensor-flow but also manually tuned the parameters to achieve acceptable accuracy for malware detection. The CNN-based deep learning model was trained without the need for manually sketched PEs before being fine-tuned, which is one of its distinctive features. Now, this proposed model can be used as a transfer learning without manually tuning any parameters for malware detection purposes because the RGB malware dataset and model parameters problems are almost solved. The proposed fine-tuned CNN model and dataset achieved an accuracy of 98.7% and a rate of error of 1.3% and performed better than other datasets and models. Accuracy can be improved by using a balanced dataset and using more variant attention layers for selecting the main feature, but it requires more computation power, which increases the cost. It is observed that in a balanced dataset of high-resolution RGB three-channel images, horizontal flip augmentation can extract more features than in a one-channel grayscale image dataset. Fitting the model with RGB and grayscale images requires fine-tuned hyper-parameters for malware identification. For ease of detection and evaluation results, the best-

performing tuned DL model among the various evaluated ANN, DNN, and DL models with grayscale and RGB image datasets is presented in detail along with the confusion matrix (CM), accuracy, precision, recall, loss, and other estimated assessment parameters. A problematical algorithm study required dataset CNN algorithm execution speed and needs for storage, which is also remarkable. This comparative analysis demonstrates the superior implementation of the author's proposed automated image-based DL model to detect malware assaults compared to modern and traditional deep learning as well as machine learning algorithms. The resilient cyber systems versus cyber-attacks, the proposed symmetrical deep learning MD techniques will help cyber security analytics improve malware detection and mitigation. It can retrieve more features and get better results in the size of large images.

When a new piece of malware released which is not included in the dataset, there are possibilities for detecting it from the augmented image dataset because the behaviour of the image is the same. Because it is known that it is possible to get new features when expanding an image, such as human face augmentation and rotation, it can see various sides of a face, but this technique may increase computation costs. It is known that CNN works like a human eye, so if applying images to all the augmentation techniques such as rotation, colour, and image size, generating a novel shape and attributes of the malware image feature is possible to extract. As long as the researcher does not get new malware samples, this technique may help anti malware software to detect new malware. Future research will also use grayscale and RGB datasets in parallel CNN at the same time, which may extract more features from four channels of the image dataset. Then, these parallel CNN models will be an ensemble, or it is possible to add more options, such as adding a variation auto encoder in the layer with DNN or parallel CNN will apply depending upon families of malware.

The limitation of this research is the need for more availability of current datasets of malware images (grayscale, RGB) on the Internet, such as file-less malware datasets (.js) through deep learning, which requires more currently working active malware. The size, or the number of malware images, is not the same for each family to train in a forward and backward pass, which affects one of the most significant hyper-parameters, which is batch size. Larger batch sizes do not achieve excellent accuracy but require more computation time, and both the learning rate and the optimizer being utilized will have a big influence. The proposed network can train more effectively if the learning rate and batch size are minimized, particularly when finally fine-tuning hyper-parameters. However, the families of multiclass malware sizes are always different by nature, and researchers are required to use augmentation to solve the limitation.

Furthermore, small datasets may cause overfitting problems, in which deep learning models work well on training data but need to be more adequate on untested data. However, research is still required to address the limitations of methods to increase data effectiveness, such as partially supervised learning, active learning, and few-shot learning. Training and executing the fine-tuned multilayer model uninterruptedly (without electricity down) might require powerful GPUs, TPUs, or clusters of processors according to their size and complexity. For that reason, the researcher used the Google Colab paid version. Making sure that deep-learning models have sufficient, high-quality data to train them is another barrier to scaling them up. Identifying the ideal hyperparameters and model optimization techniques is another aspect of scaling up deep learning models' challenging task. It might also be challenging to determine the ideal model combination due to the exponential growth in the quantity and variety of hyperparameters and algorithms for optimization. In the future, if researchers use a parallel model, speed up the training process, which is based on both synchronous and asynchronous parallel approaches of CNN and which gives more chance for adding more convolution, pooling, variation autoencoder layers, with grayscale and RGB datasets but computation done in parallel. Apply supervised and unsupervised machine learning to create a hybrid deep

learning model that can detect malware that is already present in the dataset and also predict that malware has not yet been discovered.

Acknowledgement
The authors would like to express their gratitude to Ministry of Higher Education (MOHE) and Universiti Sains Islam Malaysia (USIM) for the support and facilities provided. This research was funded by the Ministry of Higher Education (MOHE) of Malaysia under the Fundamental Research Grant Scheme (FRGS/1/2024/ICT07/USIM/01/1).

References

[1] Gulatas, Ibrahim, H. Hakan Kilinc, A. Halim Zaim, and M. Ali Aydin. "Malware threat on edge/fog computing environments from Internet of things devices perspective." *IEEE Access* 11 (2023): 33584-33606. https://doi.org/10.1109/ACCESS.2023.3262614

[2] Islam, Chowdhury Sajadul, and Md Sarwar Hossain Mollah. "A novel idea of malaria identification using Convolutional Neural Networks (CNN)." In *2018 IEEE-EMBS Conference on Biomedical Engineering and Sciences (IECBES)*, pp. 7-12. IEEE, 2018. https://doi.org/10.1109/IECBES.2018.8626669

[3] Islam, Chowdhury Sajadul, and Sarwar Hossain Mollah. "The X-Layer Optimization in CRN Using Deep Q-Network for Secure High Speed Communication." In *2019 11th International Conference on Information Technology and Electrical Engineering (ICITEE)*, pp. 1-6. IEEE, 2019. https://doi.org/10.1109/ICITEED.2019.8929997

[4] Smmarwar, Santosh K., Govind P. Gupta, and Sanjay Kumar. "Deep malware detection framework for IoT-based smart agriculture." *Computers and Electrical Engineering* 104 (2022): 108410. https://doi.org/10.1016/j.compeleceng.2022.108410

[5] Asam, Muhammad, Shaik Javeed Hussain, Mohammed Mohatram, Saddam Hussain Khan, Tauseef Jamal, Amad Zafar, Asifullah Khan, Muhammad Umair Ali, and Umme Zahoora. "Detection of exceptional malware variants using deep boosted feature spaces and machine learning." *Applied Sciences* 11, no. 21 (2021): 10464. https://doi.org/10.3390/app112110464

[6] Conti, Mauro, Shubham Khandhar, and P. Vinod. "A few-shot malware classification approach for unknown family recognition using malware feature visualization." *Computers & Security* 122 (2022): 102887. https://doi.org/10.1016/j.cose.2022.102887

[7] Akhtar, Muhammad Shoaib, and Tao Feng. "Evaluation of machine learning algorithms for malware detection." *Sensors* 23, no. 2 (2023): 946. https://doi.org/10.3390/s23020946

[8] Shaukat, Kamran, Suhuai Luo, and Vijay Varadharajan. "A novel deep learning-based approach for malware detection." *Engineering Applications of Artificial Intelligence* 122 (2023): 106030. https://doi.org/10.1016/j.engappai.2023.106030

[9] Akhtar, Muhammad Shoaib, and Tao Feng. "Malware analysis and detection using machine learning algorithms." *Symmetry* 14, no. 11 (2022): 2304. https://doi.org/10.3390/sym14112304

[10] Alnajim, Abdullah M., Shabana Habib, Muhammad Islam, Rana Albelaihi, and Abdulatif Alabdulatif. "Mitigating the Risks of Malware Attacks with Deep Learning Techniques." *Electronics* 12, no. 14 (2023): 3166. https://doi.org/10.3390/electronics12143166

[11] Djenna, Amir, Ahmed Bouridane, Saddaf Rubab, and Ibrahim Moussa Marou. "Artificial intelligence-based malware detection, analysis, and mitigation." *Symmetry* 15, no. 3 (2023): 677. https://doi.org/10.3390/sym15030677

[12] Wolsey, Adam. "The State-of-the-Art in AI-Based Malware Detection Techniques: A Review." *arXiv preprint arXiv:2210.11239* (2022).

[13] Liu, Kaijun, Shengwei Xu, Guoai Xu, Miao Zhang, Dawei Sun, and Haifeng Liu. "A review of android malware detection approaches based on machine learning." *IEEE access* 8 (2020): 124579-124607. https://doi.org/10.1109/ACCESS.2020.3006143

[14] Naeem, Hamad, Farhan Ullah, Muhammad Rashid Naeem, Shehzad Khalid, Danish Vasan, Sohail Jabbar, and Saqib Saeed. "Malware detection in industrial internet of things based on hybrid image visualization and deep learning model." *Ad Hoc Networks* 105 (2020): 102154. https://doi.org/10.1016/j.adhoc.2020.102154

[15] Awan, Mazhar Javed, Osama Ahmed Masood, Mazin Abed Mohammed, Awais Yasin, Azlan Mohd Zain, Robertas Damaševičius, and Karrar Hameed Abdulkareem. "Image-based malware classification using VGG19 network and spatial convolutional attention." *Electronics* 10, no. 19 (2021): 2444. https://doi.org/10.3390/electronics10192444

[16] Sharma, Osho, Akashdeep Sharma, and Arvind Kalia. "Windows and IoT malware visualization and classification with deep CNN and Xception CNN using Markov images." *Journal of Intelligent Information Systems* 60, no. 2 (2023): 349-375. https://doi.org/10.1007/s10844-022-00734-4

[17] Yadav, Pooja, Neeraj Menon, Vinayakumar Ravi, Sowmya Vishvanathan, and Tuan D. Pham. "A two-stage deep learning framework for image-based Android malware detection and variant classification." *Computational Intelligence* 38, no. 5 (2022): 1748-1771. https://doi.org/10.1111/coin.12532
[18] Obaidat, Islam, Meera Sridhar, Khue M. Pham, and Phu H. Phung. "Jadeite: A novel image-behavior-based approach for java malware detection using deep learning." *Computers & Security* 113 (2022): 102547. https://doi.org/10.1016/j.cose.2021.102547
[19] Chaganti, Rajasekhar, Vinayakumar Ravi, and Tuan D. Pham. "Deep learning based cross architecture internet of things malware detection and classification." *Computers & Security* 120 (2022): 102779. https://doi.org/10.1016/j.cose.2022.102779
[20] Falana, Olorunjube James, Adesina Simon Sodiya, Saidat Adebukola Onashoga, and Biodun Surajudeen Badmus. "Mal-Detect: An intelligent visualization approach for malware detection." *Journal of King Saud University-Computer and Information Sciences* 34, no. 5 (2022): 1968-1983. https://doi.org/10.1016/j.jksuci.2022.02.026
[21] Zhu, Huijuan, Huahui Wei, Liangmin Wang, Zhicheng Xu, and Victor S. Sheng. "An effective end-to-end android malware detection method." *Expert Systems with Applications* 218 (2023): 119593. https://doi.org/10.1016/j.eswa.2023.119593
[22] Venkatraman, Sitalakshmi, Mamoun Alazab, and R. Vinayakumar. "A hybrid deep learning image-based analysis for effective malware detection." *Journal of Information Security and Applications* 47 (2019): 377-389. https://doi.org/10.1016/j.jisa.2019.06.006
[23] Kamboj, Akshit, Priyanshu Kumar, Amit Kumar Bairwa, and Sandeep Joshi. "Detection of malware in downloaded files using various machine learning models." *Egyptian Informatics Journal* 24, no. 1 (2023): 81-94. https://doi.org/10.1016/j.eij.2022.12.002
[24] Alazzam, Hadeel, Ahmad Sharieh, and Khair Eddin Sabri. "A lightweight intelligent network intrusion detection system using OCSVM and Pigeon inspired optimizer." *Applied Intelligence* 52, no. 4 (2022): 3527-3544. https://doi.org/10.1007/s10489-021-02621-x
[25] Cardoso, Victor Gustavo Kelis, and Ronei Jesus Poppi. "Cleaner and faster method to detect adulteration in cassava starch using Raman spectroscopy and one-class support vector machine." *Food Control* 125 (2021): 107917. https://doi.org/10.1016/j.foodcont.2021.107917
[26] Alazzam, Hadeel, Ahmad Sharieh, and Khair Eddin Sabri. "A lightweight intelligent network intrusion detection system using OCSVM and Pigeon inspired optimizer." *Applied Intelligence* 52, no. 4 (2022): 3527-3544. https://doi.org/10.1007/s10489-021-02621-x
[27] Qi, Ruobin, Craig Rasband, Jun Zheng, and Raul Longoria. "Detecting cyber attacks in smart grids using semi-supervised anomaly detection and deep representation learning." *Information* 12, no. 8 (2021): 328. https://doi.org/10.3390/info12080328
[28] Zhao, Yong-Ping, Gong Huang, Qian-Kun Hu, and Bing Li. "An improved weighted one class support vector machine for turboshaft engine fault detection." *Engineering Applications of Artificial Intelligence* 94 (2020): 103796. https://doi.org/10.1016/j.engappai.2020.103796
[29] Khraisat, Ansam, Iqbal Gondal, Peter Vamplew, Joarder Kamruzzaman, and Ammar Alazab. "Hybrid intrusion detection system based on the stacking ensemble of c5 decision tree classifier and one class support vector machine." *Electronics* 9, no. 1 (2020): 173. https://doi.org/10.3390/electronics9010173
[30] Nguyen, Quoc Thong, Kim Phuc Tran, Philippe Castagliola, Truong Thu Huong, Minh Kha Nguyen, and Salim Lardjane. "Nested one-class support vector machines for network intrusion detection." In *2018 IEEE Seventh International Conference on Communications and Electronics (ICCE)*, pp. 7-12. IEEE, 2018. https://doi.org/10.1109/CCE.2018.8465718
[31] Min, Byeongjun, Jihoon Yoo, Sangsoo Kim, Dongil Shin, and Dongkyoo Shin. "Network anomaly detection using memory-augmented deep autoencoder." *IEEE Access* 9 (2021): 104695-104706. https://doi.org/10.1109/ACCESS.2021.3100087
[32] Verkerken, Miel, Laurens D'hooge, Tim Wauters, Bruno Volckaert, and Filip De Turck. "Towards model generalization for intrusion detection: Unsupervised machine learning techniques." *Journal of Network and Systems Management* 30 (2022): 1-25. https://doi.org/10.1007/s10922-021-09615-7
[33] Mahfouz, Ahmed M., Abdullah Abuhussein, Deepak Venugopal, and Sajjan G. Shiva. "Network intrusion detection model using one-class support vector machine." In *Advances in Machine Learning and Computational Intelligence: Proceedings of ICMLCI 2019*, pp. 79-86. Springer Singapore, 2021. https://doi.org/10.1007/978-981-15-5243-4_7
[34] Binbusayyis, Adel, and Thavavel Vaiyapuri. "Unsupervised deep learning approach for network intrusion detection combining convolutional autoencoder and one-class SVM." *Applied Intelligence* 51, no. 10 (2021): 7094-7108. https://doi.org/10.1007/s10489-021-02205-9

[35] Jamaluddin, Khairul Rohaizzat, and Shafaf Ibrahim. "A Review on Occluded Object Detection and Deep Learning-Based Approach in Medical Imaging-Related Research." *Journal of Advanced Research in Applied Sciences and Engineering Technology* 34, no. 2 (2023): 363-373. https://doi.org/10.37934/araset.34.2.363373